Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1 (currently amended): A method of managing an input buffer in a media player for playing a media file, the media file comprising a stream of frames, each frame having at least a main_data field containing encoded media samples and a main_data_begin field indicating an overflow of the main_data field, the media player including a parser, an input buffer, a decoder, and a totalizer, the parser is capable of parsing the stream of frames to the decoder and informing the decoder whether to decode from the beginning of the media file, or from the middle of the media file, the method comprising:

if the decoder is informed to decode from the middle of the media file, then:

locating a first frame having a first main_data_begin field and a first main_data field, if a value in the totalizer is less than a value in the first main_data_begin field, adding a size of the first main_data field to the totalizer, and storing the first main_data field in the input buffer; and

locating a second frame which is downstream to the first frame, the second frame having a second main_data_begin field and a second main_data field, if a value in the totalizer is equal to or larger than a value in the second main_data_begin field, decoding the stream of frames starting from the second frame using both the first main_data field stored in the input buffer and the second main_data field; and

if the decoder is informed to decode from the beginning of the media file, then locating a third frame having a third main_data_begin field with a value of zero and a third main_data field, and decoding the stream of frames starting from the third frame.

Claim 2 (original): The method of claim 1 wherein the media file is an MP3 file.

Claim 3 (original): The method of claim 1 wherein the totalizer is initialized to zero.

Claim 4 (currently amended): A method of managing an input buffer in a media player for playing a media file, the media file comprising a stream of frames, each frame having at least a main_data field containing encoded media samples and a main_data_begin field indicating an overflow of the main_data field, the media player including a totalizer and an input buffer, the method comprising:

locating a first frame having a first main_data_begin field and a first main_data field, if a value in the totalizer is less than a value in the first main_data_begin field, adding a size of the first main_data_field to the totalizer, and storing the first main_data field in the input buffer; and

locating a second frame which is downstream to the first frame, the second frame having a second main_data_begin field and a second main_data field, if a value in the totalizer is equal to or larger than a value in a second main_data_begin field, decoding the stream of frames starting from the second frame using both the first main_data field stored in the input buffer and the second main_data field.

Claim 5 (original): The method of claim 4 wherein the media file is an MP3 file.

Claim 6 (original): The method of claim 4 wherein the totalizer is initialized to zero.

Claim 7 (currently amended): A method of managing an input buffer of a playback control for playing an MP3 (Motion Pictures Experts Group Layer III Audio) file on an MP3 player, the MP3 file comprising a sequential series of frames containing data, the method comprising:

locating a first ~~downstream~~ frame, if a value in a totalizer is less than a value in a main_data_begin field of the first frame, adding a calculated size of a main_data of the first frame to the totalizer, ~~and~~

storing the main_data of the first frame in [[the]] an input buffer for later referencing[[.]]; and

locating a second frame which is downstream to the first frame, if a value in the totalizer is equal to or larger than a value in a main_data_begin field of the second frame, decoding the stream of frames starting from the second frame using both the first main_data field stored in the input buffer and a main_data field of the second frame.

Claim 8 (original): The method of claim 7 further comprising reading an error check field if the error check field is present in the frame, and using the error check field to verify integrity of data within the frame.

Claim 9 (original): The method of claim 7 wherein the input buffer comprises a memory accessible by the playback control.

Claim 10 (original): The method of claim 7 wherein the playback control selects a parser or the playback control selects the input buffer as a source of audio data to be processed and played.

Claim 11 (original): The method of claim 7 wherein the totalizer is initialized to zero.

Claim 12 (original): The method of claim 7 further comprising using a variable to indicate that a starting frame has been located.

Claim 13 (original): The method of claim 12 wherein the variable is of a Boolean type.

Claim 14 (original): The method of claim 7 further comprising decoding a header of the frame.

Claim 15 (original): The method of claim 7 further comprising decoding an audio data of the frame.

Claim 16 (original): The method of claim 7 further comprising locating a synchronization word of the frame.

Claims 17-19 (cancelled):